PDASet Software
http://www.pdaset.com

PosPrintPro ver. 1.0.0 Library
User Manual

Introduction 2
Connection options and used hardware 3
  Matrix printer connected via COM interface 3
  An ink-jet printer with built - in IrDA interface 4
  A matrix printer connected by means of IrDA-COM adapter 4
  A matrix printer connected via a cradle with built-in IrDA-COM adapter 5
  Usage of solution POSPack 6
PosPrintPro API 7
  Functions 7
    ppVersion 7
    printInstance 7
  Interfaces and Classes 7
    ppPrint 7
    ppDocument 10
    ppContext 12
  Structures 14
    Settings 14
    rawPrinterSettings 15
Examples of usage PosPrintPro API 18

Introduction

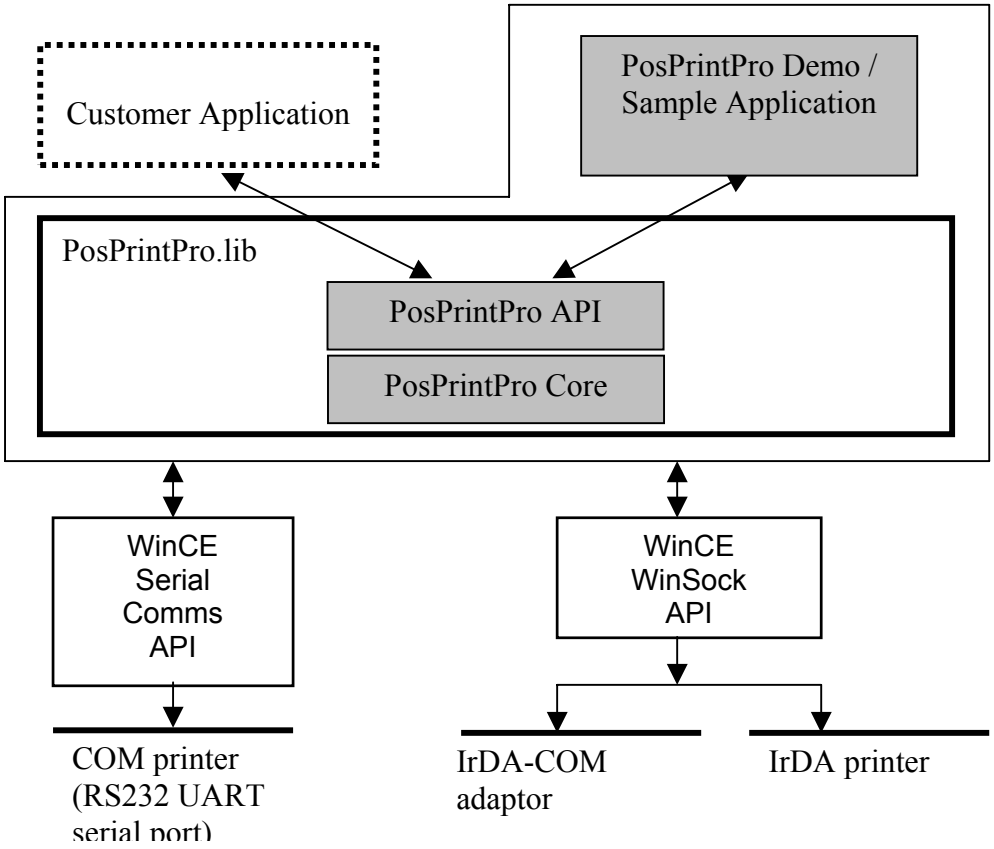
PosPrintPro it an easy-to-use tool of a software developer, allowing one to add printing capability (on matrix and POS printers) for hand-helds operating under Pocket PC /Pocket PC 2002 operating systems.

Key features of ***PosPrintPro*** library which favorably distinguish it from other libraries of the similar schedule and make it more lucrative, are:

- Simple-in-use library engine giving to the designer a print-out capability of elaborating documents with text complex formatting, by breaking documents into the main components, such as header and footer, introduction and conclusion sections, cyclical-output data in the document body, etc.
- Simplicity of connection and usage of the library in your applications.

Adding a print preview option for documents prepared for print-out.
Printing of any documents, such as invoices, receipts, load reports and others.
Possibility of creating your own document templates for their further usage in your applications.
Operation with matrix printers that allows one a fast and easy print-out of up to 5 full-page copies of the same document in "field" on-the-go conditions.

PosPrintPro is a program library written in C language which allows the user's application to execute print preview and printout of documents on IBM- and Epson-compatible printers connected to a Pocket PC via COM or infrared (IrDA) interfaces.



PosPrintPro it is the library unit and the initial code in C++ language, developed for Pocket PC applications and supporting the Pocket of PC and Pocket PC 2002 operating systems.

Connection options and used hardware

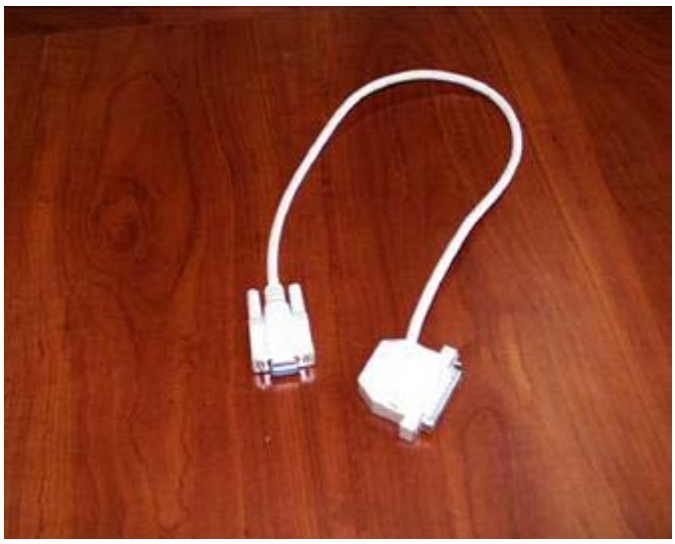
PosPrintPro can be used in the various configurations and different hardware. For the sake of brevity, only few options of usage of the given library with various equipment are listed below.

Matrix printer connected via COM interface

Let's discuss the two main options for Pocket PC connection to a standard matrix printer with COM the interface:

1. A pocket computer with a COM port is connected directly to the printer with usage of a special cable. We shall illustrate the given connection type by the example of handheld computer Compaq iPAQ and printer Epson LX 300 +

The required cable for the given connection can be ordered directly from our company.



2. By using a standard cable, printer is connected to COM port of Pocket PC cradle. We shall illustrate the given connection type by an example of handheld computer Rover PC P5 and printer OKI Microline 280.

For the given sort of connection in ***PosPrintPro*** library the type of printer PRINTER_COM is used.

An ink-jet printer with built - in IrDA interface

The pocket computer, with help of IrDA interface is connected directly to the printer with built - in IrDA interface. We shall illustrate the given connection type by an example of handheld computer Dell Axim 5 and printer Canon BJC-85.

For the given type of connection in ***PosPrintPro*** library the type of printer PRINTER_IRDA_CLIENT is normally used.

A matrix printer connected by means of IrDA-COM adapter

IrDA adapter with built-in support of conversion for RS232 port into IrDA (which does not require any software for usage of a special IrDA protocol) is connected to COM port of a standard matrix printer. The pocket PC with help of IrDA interface is connected to IrDA-COM adapter. We shall illustrate the given connection type by the example of handheld computer Casio Cassiopeia E-125, IrDA-COM ACTiSYS ACT-IR100S adapter and matrix printer OKI Microline 280.

For the given sort of connection in ***PosPrintPro*** library the type of printer PRINTER_IRDA_CLIENT is normally used.

A matrix printer connected via a cradle with built-in IrDA-COM adapter

The printer is connected by a standard cable to COM port on a Pocket PC cradle, which, in its turn, is connected to the respective Pocket PC through IrDA interface. We shall illustrate the given connection type by an example of handheld computer Casio Cassiopeia IT-70M30E and printer Epson LX 300 +.

For the given type of connection in ***PosPrintPro*** library the type of printer PRINTER_IRDA_SERVER (as in the example considered above), and PRINTER_IRDA_CLIENT for other type of equipment can be used.

Usage of solution POSPack

PosPrintPro can also be applied within framework a ready-made solution POSPack represented on the site http://www.pospack.com/.

For the given sort of connection in ***PosPrintPro*** library the type of printer PRINTER_COM is used.

PosPrintPro API

Functions

ppVersion

ppVersion function returns number of the current version of ***PosPrintPro*** library.

DWORD ppVersion(VOID);

Parameters
This function has no parameters.

Return value
If the function is executed successfully, the return value such as **DWORD,** contains high and low numbers of versions of the library.

printInstance

PrintInstance function returns the address to the base interface of a current copy of the library.

ppPrint* printInstance(VOID);

Parameters
This function has no parameters.

Return value
If the function is executed successfully, returned value contains the pointer on the base interface of ***PosPrintPro*** library - **ppPrint** class.
If the function is executed unsuccessfully, the returned value is equal to **NULL**.

Interfaces and Classes

ppPrint

ppPrint it is the base interface of ***PosPrintPro*** library providing access to the main functionalities of printer parameters' setup and document processing.

Methods
setup
setup method installs current customizations of the printer.

bool setup(const ppPrint::Settings* settings);

Parameters
settings
Address of the structure containing settings of the printer

Return value
If the method is executed successfully, return value is not equal to zero.
If the method is executed unsuccessfully, return value is equal to zero.

settings

settings method returns current printer settings.

```
const ppPrint::Settings& settings();
```

Parameters
This method has no parameters.
Return value
The method returns current printer settings.

rawPrint

rawPrint method provides printing of the text generated by application-defined callback function.

```
bool rawPrint ( void (*callback)( ppContext* ) );
```

Parameters
callback
The address of the application-defined callback function providing line-by-line text generation.
Return value
If the method is executed successfully, return value is not equal to zero.
If the method is executed unsuccessfully, return value is equal to zero.
Remarks
Within the framework of the **rawPrint** method, control on the formed text pagination (i.e., breakdown by pages) is user-defined and executed by the callback function, specified by the transferred address.

rawPreview

rawPreview method provides preview of the text generated by application-defined callback function.

```
bool rawPreview( void (*callback)( ppContext* ) );
```

Parameters
callback
The address of the application-defined callback function providing line-by-line text generation.
Return value
If the method is executed successfully, return value is not equal to zero.
If the method is executed unsuccessfully, return value is equal to zero.
Remarks
Within the framework of the **rawPreview** method, control on the formed text pagination (i.e., breakdown by pages) is user-defined and executed by the callback function, specified by the transferred address.

print

print method provides printing of the document formed by the **document** class derived from **ppDocument** base class.

bool print(ppDocument* document);

Parameters
document
The address of the class copy derived from **ppDocument** base class which provides creation of the text of the document.
Return value
If the method is executed successfully, return value is not equal to zero.
If the method is executed unsuccessfully, return value is equal to zero.
Remarks
Within the framework of the **print** method, control on the formed text pagination is carried out by the internal resources of **PosPrintPro** library.

preview
preview method provides preview of the document formed by the **document** class derived from **ppDocument** base class.

bool preview(ppDocument* document);

Parameters
document
The address of the class copy derived from **ppDocument** base class which provides creation of the text of the document.
Return value
If the method is executed successfully, return value is not equal to zero.
If the method is executed unsuccessfully, return value is equal to zero.
Remarks
Within the framework of the **preview** method, control on the formed text pagination is carried out by the internal resources of **PosPrintPro** library.

batchPrint
batchPrint method provides batch printing of documents generated by the classes derived from **ppDocument** base class.

bool batchPrint(ppDocument** document, unsigned qty);

Parameters
document
The address of the array, each unit of which is a copy of a class derived from **ppDocument** base class and which provides creation of the document's text.
qty
Number of the array units transferred with the *document* address.
Return value
If the method is executed successfully, return value is not equal to zero.
If the method is executed unsuccessfully, return value is equal to zero.
Remarks
Within the framework of the **batchPrint** method, control on the formed text pagination is carried out by the internal resources of **PosPrintPro** library.

batchPreview

batchPreview method provides batch preview of the documents generated by the classes derived from **ppDocument** base class.

bool batchPreview(ppDocument** document, unsigned qty);

Parameters

document

The address of the array, each unit of which is a copy of a class derived from **ppDocument** base class and which provides creation of the document's text.

qty

Number of the array units transferred with the *document* address.

Return value

If the method is executed successfully, return value is not equal to zero.
If the method is executed unsuccessfully, return value is equal to zero.

Remarks

Within the framework of the **batchPreview** method, control on the formed text pagination is carried out by the internal resources of **PosPrintPro** library.

lastError

lastError method returns last error code value of the current copy of the ***PosPrintPro*** library.

ppPrint::Error lastError();

Parameters

This method has no parameters.

Return value

The return value is the current copy of the library last error code value.

ppDocument

ppDocument it is the base class of the document of **PosPrintPro** library, which provides the main functionality on creation of the document's text.

Members

m_documentDisplayedInfo

The textual line containing the information on the document, which line is shown in the preview and print control window for visual identification of the processed document.

m_rowQty

Number of cyclic parts (e.g. table lines) which are presented in a body of the document.

Methods

firstPageHeader

firstPageHeader method is called out by **PosPrintPro** library, for creation of the text of the header of the first page of the document.

void firstPageHeader(class ppContext* context);

Parameters

context

The address of the copy of context class for the device to which the document's text output is performed.

Return value

The method has no return values.

pageHeader

pageHeader method **pageHeader** is called out by **PosPrintPro** library for header text creation in the subsequent pages of the document.

void pageHeader(class ppContext* context);

Parameters

context

The address of the copy of context class for the device to which the document's text output is performed.

Return value

The method has no return values.

pageFooter

pageFooter method is called out by **PosPrintPro** library for footer text creation in all pages of the document, except the last one.

void pageFooter(class ppContext* context);

Parameters

context

The address of the copy of context class for the device to which the document's text output is performed.

Return value

The method has no return values.

lastPageFooter

lastPageFooter method is called out by **PosPrintPro** library, for creation of the footer text in the last page of the document.

void lastPageFooter(class ppContext* context);

Parameters

context

The address of the copy of context class for the device to which the document's text output is performed.

Return value

The method has no return values.

title

title method is called out by **PosPrintPro** library, for text generation of the document's title (or the initial area of the document).

void title(class ppContext* context);

Parameters

context

The address of the copy of context class for the device to which the document's text output is performed.

Return value

The method has no return values.

summary

summary method is called out by **PosPrintPro** library, for creation of the text of the document's summary (or the final area of the document).

void summary(class ppContext* context);

Parameters

context

The address of the copy of context class for the device to which the document's text output is performed.

Return value

The method has no return values.

row

row method is called out by **PosPrintPro** library, for creation of the text of cyclic parts (e.g. table lines) presented in a body of the document.

void row (class ppContext* context);

Parameters

context

The address of the copy of context class for the device to which the document's text output is performed.

Return value

The method has no return values.

ppContext

ppContext it is the base interface of devices of **PosPrintPro** library, providing main functionality on the text output to the device.

Methods

create

create method initializes context current ***PosPrintPro*** library device, for example, installs connection to the printer.

bool create ();

Parameters

This method has no parameters.

Return value

If the method is executed successfully, return value is not equal to zero.
If the method is executed unsuccessfully, return value is equal to zero.

destroy

destroy method closes context current ***PosPrintPro*** library device, for example, closes connection to the printer.

void destroy ();

Parameters
This method has no parameters.
Return value
The method has no return values.

startDocument
startDocument method initializes the printer and transferred it into a ready-to-print mode.

void startDocument ();

Parameters
This method has no parameters.
Return value
The method has no return values.
Remarks
The method uses a Control sequence **pageSetupSequence**, current installations of the printer.

endDocument
endDocument method completes printing the document.

void endDocument ();

Parameters
This method has no parameters.
Return value
The method has no return values.

startPage
startPage method transferred the printer into a ready-to-print mode for printing new page.

void startPage ();

Parameters
This method has no parameters.
Return value
The method has no return values.
Remarks
The method uses a Control sequence **newPageSequence**, current installations of the printer.

endPage
endPage method completes printing of the current page. For printing a new page, it is necessary to call **startPage** method.

void endPage ();

Parameters
This method has no parameters.
Return value
The method has no return values.
Remarks
The method uses a Control sequence **endPageSequence**, current installations of the printer.

line
line method provides output of the text line to the device, using control sequences (or escape sequences) of current printer settings for formatting and switching to the next line.

void line (const char* str);

Parameters
str
Address of the string ended with a null character, which is printed out.
Return value
The method has no return values.
Remarks
The method uses the Control sequence **lineSequence** before output of line to the device, and **newLineSequence** after output of the line, with current settings of the printer.

currentLine
currentLine method returns number of current line on the page.

unsigned currentLine ();

Parameters
This method has no parameters.
Return value
Return value is equal to the number of the current line on page.

currentPage
currentPage method returns number of the current page of the output document.

unsigned currentPage ();

Parameters
This method has no parameters.
Return value
Returned value is equal to number of current page of the output document.

Structures

Settings

Contains the information used for the printer setup, as well as for the document formatting and writing the message on the document printout termination. It is used by **setup** and **settings** methods of **ppPrint** interface.

```
typedef struct Settings {
   PrinterType          printerType;
   unsigned             rowsPerPage;
   unsigned             charactersPerRow;

   struct {
      bool              waitForCommit;
      TCHAR             statusMessage[ MAX_STATUS_MESSAGE_LENGTH + 1];
   } controllerSettings;

   rawPrinterSettings   *rawPrinter;

} Settings;
```

Members

printerType

The used printer type.

At present, only the following three types of printers are supported:

PRINTER_COM	Printer is connected to a handheld (Pocket PC) via COM interface.
PRINTER_IRDA_CLIENT	Printer is connected to a handheld (Pocket PC) via IrDA interface. Connection with remote IrDA device is initiated by the handheld.
PRINTER_IRDA_SERVER	Printer is connected to a handheld computer (Pocket PC) via IrDA interface. Connection with the remote IrDA is initiated by the remote device.

rowsPerPage

Number of lines on page of the document.

charactersPerRow

Number of output characters in the document's line.

waitForCommit

The flag indicating the requirement of screen visualization of the dialogue on printout termination.

statusMessage

String ending with a null character which is output in the dialogue of the printout termination if the value *waitForCommit* is not equal to zero.

rawPrinter

Structure that contains the information used for customization, initialization and handle of the printer.

rawPrinterSettings

Contains the information used for customization, initialization and handle of the printer.

```
typedef struct {
   char          portName[MAX_PORT_NAME_LENGTH + 1];
   unsigned      lineOffsetLength;
```

```
    char        pageSetupSequence[MAX_SEQUENCE_LENGTH];
    unsigned    pageSetupSequenceLength;

    char        newPageSequence[MAX_SEQUENCE_LENGTH];
    unsigned    newPageSequenceLength;

    char        endPageSequence[MAX_SEQUENCE_LENGTH];
    unsigned    endPageSequenceLength;

    char        lineSequence[MAX_SEQUENCE_LENGTH];
    unsigned    lineSequenceLength;

    char        newLineSequence[MAX_SEQUENCE_LENGTH];
    unsigned    newLineSequenceLength;
} rawPrinterSettings;
```

Members

portName

Name of a port, to which the printer is connected, for example “COM1:”. For the printers connected through infrared (IrDA) interface, a name of a port corresponds to “IrDA:IrCOMM”.

lineOffsetLength

Size of the left indent for each line (i.e. left offset length of the line), in characters.

pageSetupSequence

Control sequence of initialization of the printer. The given sequence is used by **startDocument** method **ppContext** interface.

pageSetupSequenceLength

Length of the control sequences of initialization of the printer in bytes.

newPageSequence

Control sequence of initialization of a new page. The given sequence is used by **startPage** method **ppContext** interface.

newPageSequenceLength

Length of the control sequences of initialization of a new page in bytes.

endPageSequence

Control sequence of he current page print-out completion. The given sequence is used by **endPage** method **ppContext** interface.

endPageSequenceLength

Length of the control sequence of the current page print-out completion in bytes.

lineSequence

Control sequence on the format setting for the output line. The given sequence is used by the **line** method **ppContext** interface.

lineSequenceLength

Length of a control sequence on the format setting for the output line in bytes.

newLineSequence
Control sequence of word wrap-around. The given sequence is used by **line** method **ppContext** interface.

newLineSequenceLength
Length of the control sequences of word wrap-around in bytes.

Examples of usage PosPrintPro API

Examples of connection and usage ***PosPrintPro*** library in the application are in folder **DEMO** and represent the following:

1. ***demo1*** application is an example of a simple application of the library for printing a line “Hello world!”.
2. ***demoDLL*** application is an example on creation of templates’ library consisting of templates of two documents: a very simple document with one-line output and a multipart document with complicated formatting (invoice).
3. ***demo2*** application is an example of printing a multipart document with complicated formatting (invoice), where the template from the example ***demoDLL*** is used.
4. ***demo3*** application is an example of printing several documents in the batch mode, with usage of templates from the example ***demoDLL***.

All examples are developed with usage of the integrated developer’s environment Microsoft eMbedded Visual C ++ 3.0.